AMENDMENTS TO THE SPECIFICATION

Please amend the specification of the application as follows:

On page 10, please rewrite the paragraph commencing at line 28, as follows:

FIG. 6 illustrates a layout of the memory device 11 in the computer system 2 of FIG. 1 according to an embodiment of the invention. As briefly discussed above in the description of FIG. 1, the image file 31 is used to program the contents of the memory device 11. As shown in FIG. 6, the memory device 11 is divided into sectors 91-98 for storing BIOS program code and data. It will be appreciated that the various parts of the BIOS program code discussed above in the description of FIG. 3 may be assigned to the sectors in the memory device 11 during the BIOS build process. Here, the boot block region is stored in sector 91 (Sector 1), the main BIOS region is stored in sectors 92-96 (Sectors 2 through N-2), and the non-essential region (i.e., the non-essential blocks) is stored in sectors 97-98 (Sectors N-1 through N). It will be appreciated that the main BIOS region may be stored in any of the sectors 1-N in the memory device 11 and that the sectors storing the program code and data for the ~~man~~ main BIOS region need not be contiguous.

On page 13, please rewrite the paragraph commencing at line 19, as follows:

FIG. 8 illustrates an operational flow for updating the sectors in the memory device 11 with non-essential blocks. The logical operations 800 begin at operation 805 where the update utility 31 determines the size of a non-essential block to be programmed or mapped to the sectors of the memory device 11 reserved for non-essential blocks. As discussed above with respect to FIG. 6, the block size of each non-essential block may be stored in the header of each block. If at operation 810, the update utility 31 determines that the size of a first sector is equal to the size of a non-essential block, then the logical operations 800 continue to operation 815 where the non-essential block is mapped to the first sector in the non-essential region of the memory device. For example, if a non-essential block containing program code for displaying graphics data is 64 kilobytes and the size of the first sector is 64 kilobytes, ~~the~~ then all of the program code for displaying

the graphics data will be mapped to the first sector in the non-essential region in the memory device.

On page 14, please rewrite the paragraph commencing at line 3, as follows:

Continuing now with FIG. 8, if at operation 810, the update utility 31 determines that the size of the first sector is not equal to the size of a non-essential block, then the logical operations 800 continue to operation 820 where it is determined whether the size of the first sector is less than the size of a non-essential block. If at operation 820 it is determined that the size of the first sector is less than the size of a non-essential block, then the logical operations 800 continue to operation 825 where the non-essential block is mapped to multiple sectors in the non-essential region of the memory device. It will be appreciated that if the contents of the non-essential block completely fill one sector but fill less than a subsequent sector, then the non-essential block contents may be mapped to a paragraph multiple of the subsequent sector. If, however, at operation 820 the update utility 31 determines that the size of the first sector is greater than the size of a non-essential block, then the logical operations continue to operation 830 where the update utility maps the non-essential block contents to a portion of the first sector in the non-essential region in the memory device. It will be appreciated that the size of the portion of the first sector may be a paragraph multiple. <u>If, at operation 835, the update utility 31 determines that there are more non-essential blocks to update, then the logical operations start over at operation 805 with the next non-essential block. If this is the last non-essential block, then the update ends.</u>